Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 96 (2016) 1231 – 1239

Procedia Computer Science

20th International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2016, 5-7 September 2016, York, United Kingdom

Developing Design Support System based on Semantic of Design Model

Yoshikazu Tanaka[a*] and Kazuhiko Tsuda[b]

[a]*NARO Institute for Rural Engineering, 2-1-6, kan-nondai, Tsukuba, Ibaraki, 305-8609, Japan*
[b]*Graduate School of System and Information Engineering, University of Tsukuba, 3-29-1, Otsuka, Bunkyo, Tokyo, 112-0012, Japan*

Abstract

A design support system is developed for engineering design based on the ontology which is declarative description conceptualized the world that agents (men and programs) focused on. The design process is understood as determining the design parameters and their relationships which consists the design model. The meta-model of design model is the ontology which represented as a network in the computer system using the XML. The design model is generated from the meta-model as class in Object oriented language .The system built with the above concept provides the following abilities, 1) declarative description that engineers intend to design, 2) effective entering data into Object oriented model, 3)semantic of design model which focused by numerical simulation and graphics programed with Java. Finally, the system's validity and effectiveness is ascertained by applying it to the basic design of an irrigation pipeline.

Peer-review under responsibility of KES International

Keywords: ontology, XML, Object oriented programming, Irrigation pipeline system, Water hammer analysis,

1. Introduction

Nomenclature

a	*Pipe pressure propagation velocity (m/s)*
C^+, C^-	Trajectory of a forward wave and backward wave

* Corresponding author. Tel.: +81-29-838-7564; fax: +81-29-838-7609.
E-mail address: yokka@affrc.go.jp

Peer-review under responsibility of KES International
doi:10.1016/j.procs.2016.08.167

g	Gravitational acceleration (m/s^2)
H	Piezometric head (m)
V	Mean pipe flow velocity (m/s)
x	Axial distance of pipe (m)
t	Time elapse (s)
γ	Pipe gradient
Φ	Pipe diameter (m)

As a design support system, there are expert system, computer aided engineering (CAE), and the like. Intellectual design support systems require based on the mechanism of the expert system, not only the excellence of the architecture of the system, but also the quality of the knowledge. Configuration of the design support system consists of the inference engine around the knowledge base and the interface. Knowledges stored in the knowledge base are roughly classified into two kinds, the heuristics and the structure about the target which the engineer will design. The heuristics and the structure about the target are often used the production rule and the frame, respectively. The expert system has a biggest problem that it is difficult to objectively determine because of the rule base consisted of the strong subjective experience knowledge due to experts in the subject.

Otherwise, the reproducing a phenomenon by the numerical simulation is useful to determine objectively. CAE is a mechanism for supporting the interface to find solutions by applying discrete solution of partial differential equations for the phenomena to be analyzed. However, it is difficult for the engineers to apply numerical simulation for the case that the structure of target to be analyzed and the applicable boundary conditions are complicated with merely assistant by the graphical user interface. It is necessary that design support system has various knowledges conceptualized the structure about each targets to be analyzed for the engineer's easily utilization of the numerical simulation.

Ontology is a declarative description about the concept of the targeted world about which various agents (human or program) can understand [1, 2]. In this paper, CAE program has been developed for engineers to support designing an object based on the ontology as metadata. An irrigation pipeline system[3] is adopted as an example of the design target. Irrigation pipeline system is huge structure which the extension is so long and also the structure is so complex to understand the whole of that.

In this paper, the procedure and mechanism of its development were showed. Firstly, the declaration of the XML document for the ontology that is conceptualized the irrigation pipeline system was created. Secondly, the classes described by object oriented programming language were automatically generated using the ontology. Finally, we developed a numerical analysis program based on these classes to obtain important information on the design. Furthermore, a program that performs pre-post processing based on the ontology make it possible to assist entering data into XML file and visualization of the numerical results. As a result, we could have constructed the mechanism which once engineers describe the information of structure about the irrigation pipeline system trying to design, the numerical analysis could solve the initial and boundary conditions generated. This mechanism could help the engineers to obtain information about the size of the inner water pressure in order to determine the type of pipe in designing an irrigation pipeline system.

2. Ontology and Object Oriented Model

2.1. Method of Ontology Modelling

An irrigation pipeline consists of pipe facilities and various ancillary facilities, as shown in Table 1. This section proposes an ontology by using XML to gather various attributes of each ancillary facility in one place. The role of XML in this proposal is to represent structurally any information in various pipe and ancillary facilities that constitute an irrigation pipeline system. The schema compiler of XML makes it possible to code the class equivalent to the information in XML, so the solver can easily get information about the initial and boundary conditions from those generated instances. Additionally, it is expected to be linked with KML that has specified geographical information in order to facilitate data entry.

2.1.1 The Schema Language of XML

The frequency and order of appearance of key words enclosed by right and left brackets < > (called "tag") in a XML document file were specified using the codes shown in Table 2. The rule in XML document was conceptualized arranging hierarchically the frequency and order of tags. The computer language to specialize arranging tags is the schema language. In this study, we adopted RELAX as the schema language. RELAX (Regular Language description for XML) is a simple schema language based on the hedge automata theory for structured documents.

Table 1. Various ancillary facilities in an irrigation pipeline system

Facility Group	Examples of specific facilities
Regulating facilities	Regulating reservoir, Farm pond
Pressure control facilities	Check stand, Pressure-reducing valve, Float valve type pressure-reducing tank
Pump facilities	Water source pump, Relay pump, Booster pump, Water suction tank, Water discharge tank
Diversion facilities	Stand type division work, Closed type division work, Hydrant
Ventilation facilities	Air stick, Air valve
Protective facilities	Water hammer buffer device, Safety valve
Control facilities	Control valve, Check valve
Others	Bent pipe, Sudden contraction pipe, Sudden enlarged pipe, End pipes, Branches

Table 2. Definition of appearance frequency and order of tags

Appearance order		Appeaance frequency	
Meanig	Code	Meaning	Code
Appeaance in order	Sequence	Definitely appears once	Node
		Appears either 0 or 1 time	?
Either one appears	Choice	Appears 1 or more times	+
		Appears 0 or more times	*

2.1.2. Method of Stipulating a Pipeline in the Document

The conceptualizing ancillary facility is represented by specifying the contents of tags of the connections, locations, structures, and characteristics in Table 3. Tags express the names of information that represent each facility, and the task of defining these tags is the heuristic task first noticed when this information was needed. Consequently, the tags in Table 3 are the top-level framework established in advance, so it is possible to specialize information related to connections, locations, structures, and characteristics to conceptualize any facilities. For example, in the case of valve and bend, the bottom tags defined under the top-level tags of the connections, locations, structures, and characteristics are shown in Table 4. The bottom tags in other facilities can be also conceptualized by the same method. Ancillary facilities with more complex structures are hierarchically conceptualized by systematically combining other ancillary facilities as components. Taking the stand type water division works shown in Figure 1 as an example, the components are spillway, inlet, outlet, valve, gate, tanks and barrier wall, and those components also possess the information of connection, location, structure and characteristics.

Table 3. Content of tags in facility A

Tags in facility A	Meanig	Contents of lower tag
<ConnetionInfoOfA>	Connection information	ID of facility connected to upstream side and downstream side
<PositionInfoOfA>	Location information	3D location information about facility
<StructureInfoOfA>	Structure information	Typical dimensions and drawing of facility, etc.
<CharacterInfoOfA>	Characteristics information	Coefficient of flow velocity, coefficient of loss and other values representing characteristics

Table 4. Difference between bottom tags in the concepts of valve and bend

Valve <IS_Valve>			Bend <IS_Bend>		
Top tag	Button Tag	Contents	Top tag	Button Tag	Contents
<ConnectInfoOfValve?>	<upper?>	ID of pipe on upstream side	<ConnectInfoOfBend?>	Same as on left	
	<lower?>	ID of pipe on downstream side			
<PositionInfoOfValve?>	<displaymethod?>	Method of displaying location information	<PositionInfoOfBend?>	Same as on left	
	<longitude?>	Longitude			
	<latitude?>	Latitude			
	<altitude?>	Altitude			
	<x?>	x coordinate of 3D spaces			
	<y?>	y coordinate of 3D spaces			
	<z?>	z coordinate of 3D spaces			
<StructureInfoOfValve?>	<kindOfValve>	Type of value	<StructureInfoOfBend?>	<aAngle?>	Angle of a Reflection
	<ValveDiameter>	Internal diameter		<aLength?>	Length interval reflection
	<drawing*>	Blueprint		<innerDiameter>	Internal diameter
	<picture*>	Photograph		<timeOfRefrect>	Frequency of reflections in a Bend
<CharacterInfoOfValve?>	<triggerTime?>	Trigger time		<material>	Material
	<TimeORformula?>	Designation of method of representing aperture		<totalAngle>	Total angles in a Bend
	<time_totalOpenDegree*>	Aperture by time		<totalLength>	Total length in a Bend
	<formula_totalOpenDegree*>	Aperture by secondary equation		<drawing*>	Blueprint
	<hydraulicPressure>	Hydrostatic pressure		<picture*>	Photogpaph
	<initialOpenDegree?>	Aperture at initial condition	<CharacterInfoOfBend?>	<hydraulicPressure>	Hydraulic Pressure
				<coefficientOfFormDrag>	Coefficient of local loss

The concept of the pipeline was stipulated as the schema language ISML (Irrigation System Markup Language) for the XML document that describes the information about the irrigation pipeline. The Left side in Figure 2 shows the overall basic structure of ISML.

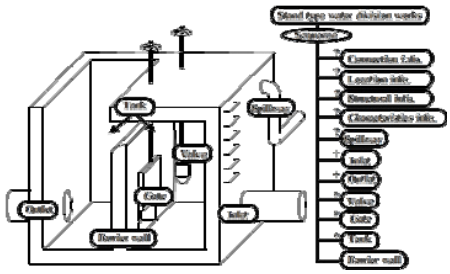

Fig. 1. Conceptual scheme showing the hierarchical structure of the stand type water division works

Fig. 2. Outline of the tag structure in ISML (Left) and ISOG formed by RELAXER (Right)

2.2. Method of Object Oriented Modelling

In object oriented modelling, at first, the abstract classes were defined by abstracting the behavior and attributes of the pipe and ancillary facilities. Then, the concrete ancillary facilities were defined as the derived class that achieves the specifications by overriding the contents of the method with the same name in the abstract class or by including another ancillary facilities class. Instances of derived class can act in the code using the polymorphism to represent complex structures and behaviors of various ancillary facilities. We designed these pipe and ancillary facilities using UML and implemented Java objects library OOWHL [4].
OOWHL is explained below. The basic unit of computation model that constitutes an irrigation pipeline is one pipe that is placed between the ancillary facilities located at upstream edges and downstream edges. The pipeline that some basic units have connected is a hydraulic unit. A long irrigation pipeline consists of multiple hydraulic units.

2.2.1. Pipe Facilities

To perform one-dimensional numerical analysis of the basic equations (1) and (2), the pipe is represented as a path on a diagram. A path is a diagram consisting of two points, a start point and an end point. Therefore, as shown in left side of Figure 3, the upstream and downstream edges are defined as Point class as explained later, and the path is defined as Line class that includes these two edges. In the Pipe class derived from Line class, an algorithm that performs numerical calculation based on the method of characteristics is implemented. The various specific pipes were derived from the Pipe class. An irrigation pipeline consists of linked instances made from those derived Pipe classes.

2.2.2. Ancillary Facilities

The derived classes representing ancillary facilities inherited Point class which have attributes such as three-dimensional space coordinate location information and fluid dynamic values etc. and methods coded function to

execute the entry data, the computation, and the output (right side in Figure 3). The methods defined to compute boundary conditions in Point class has been overridden in the derived classes of ancillary facilities in order to represent their different characteristic behavior. In order to utilize the polymorphism of each instances generated those classes, the design pattern called Factory Method was applied to generate those instances.

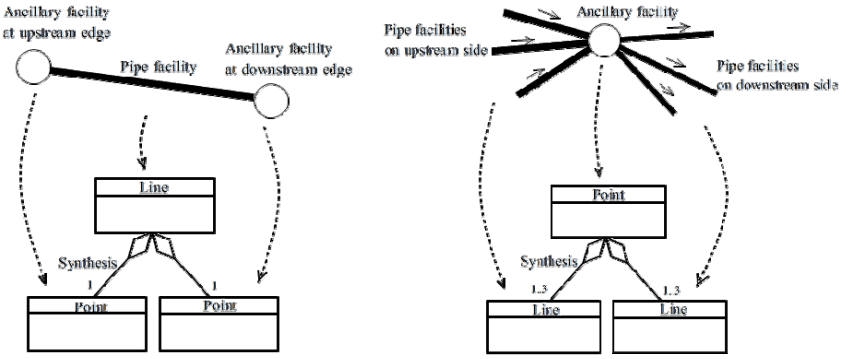

Fig. 3. The connection relationship with Point classes and a Line class

2.3. Method of Obtaining Information from the XML File

A schema compiler automatically can code classes of an object oriented program from the schema language. That feature make possible to generate object with equivalent information in the XML document. We used RELAXER[5] which has two functions as a parser and a schema compiler. RELAXER can automatically code Java classes which operate information identical to the tags defined based on stipulations described by RELAX[6]. The classes shown in the right side of Figure 2 were generated from the tags of ISML shown in the left side of Figure 2. We will call that objects library ISOG (Irrigation System's Objects Generated). The fields corresponding to the names of bottom tags and the methods to get and set the values of each fields have been coded inside each classes. The fields and methods about valve have been coded by RELAXER shown in Table 5 from the tags listed on the left side of Table 4. Because names of tag and names of domain are in a one-to-one correspondence, these methods let the solver operate value about data in the XML document from equivalent object without via DOM.

Table 5. Generated fields and methods in Valve class which is equivalent to XML for valve

Class name	Fields	Methods	Contents
ConnectInfoOfValve	Lower	getId()	Gets ID of facility connected on downstream side
	Upper	getId()	Gets ID of facility connected on upstream side
PositionInfoOfValve		getDisplaymethod()	Gets location infromation display method (geodetic reference or Euclid coordinates)
	Longitude	getContent()	Gets longitude
	Latitude	getContent()	Gets latitude
	Altitude	getContent()	Gets elevation
	X	getContent()	Gets x coordinate value in Euclid coordinates
	Y	getContent()	Gets y
	Z	getContent()	Gets z
StructureInfoOfValve		getKindOfValve()	Gets type of valve
	valveDiameter	getContent()	Gets diameter of valve
CharacterInfoOfValve	TriggerTime	getContent()	Gets that valve operation status (trigger time)
		getTimeOfFormula()	Gets valve operating time and aperture
	HydrostaticPressure	getContent()	Gets initial value of pressure head
	InitialOpenDegree	getContent()	Gets initika value of aperture of valve

3. Estimations and discussion

3.1. Example of a pipeline System

A hydraulic unit of the trunk pipeline for area N in Prefecture T of Japan was used. This hydraulic unit uses pipe of the pump pressure feed type as shown in Fig.4. The pipe length is approximate 6 km and the water level difference between the pump and the discharge chamber 30 m. There are 74 incidental facilities between the pump and chamber. By numerical analysis, we reproduced the transient hydraulic phenomenon that would occur if pumps suddenly stopped due to a loss of power.

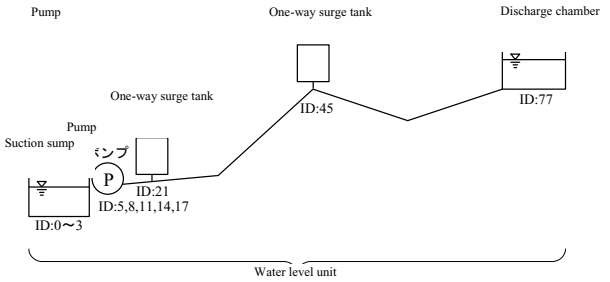

Fig. 4. Basic structure of example hydraulic unit for analysis

3.2. Numerical Analysis Technique

The basic equations of hydraulic phenomenon in pipes are as follows [7] :

$$g\frac{\partial H}{\partial x}+\frac{\partial V}{\partial t}+V\frac{\partial V}{\partial x}+\frac{fV^2}{2\Phi}=0 \tag{1}$$

$$\frac{a^2}{g}\frac{\partial V}{\partial x}+\frac{\partial H}{\partial t}+V\left(\frac{\partial H}{\partial x}+\sin\gamma\right)=0 \tag{2}$$

The method of characteristics is used to solve basic equations (3) and (4). Fig. 5 shows the mean pipe flow velocity V_M^{n+1} and pressure head H_M^{n+1} at the next time step n+1 calculated by superposing their values at two places where the two characteristic lines C^+ and C^- intersect with the computational grids of the previous step. Since the characteristic line does not pass computational grid points in most cases, the value at M must be calculated from the values of R and S from equations (3) and (4).

$$H_M^{n+1}=\frac{a}{2g}[H_R^n+H_S^n+\frac{a}{g}\left(V_R^n-V_S^n\right)-\frac{f\Delta t}{2\Phi}\left(V_R^n\left|V_R^n\right|-V_S^n\left|V_S^n\right|\right)-\frac{g\left(V_R^n+V_S^n\right)}{a}\sin\gamma] \tag{3}$$

$$V_M^{n+1}=\frac{1}{2}[V_R^n+V_S^n+\frac{g}{a}\left(H_R^n-H_S^n\right)-\frac{f\Delta t}{2\Phi}\left(V_R^n\left|V_R^n\right|+V_S^n\left|V_S^n\right|\right)-\frac{g\left(V_R^n-V_S^n\right)}{a}\sin\gamma] \tag{4}$$

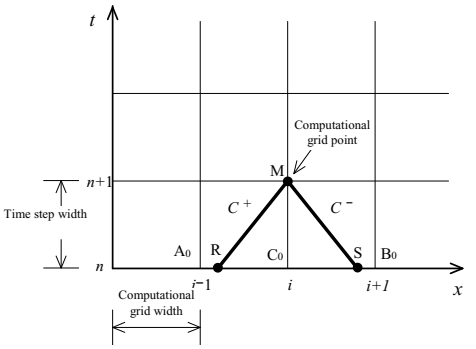

Fig. 5. Positions of characteristic lines and computational grid points

The numerical analysis program screen displays the XML files of incidental and pipe facilities created by pre-processing on the second and third lists from the left (Fig.6). Files specify the range of the hydraulic unit displayed on the left-end list with files giving other information necessary for numerical calculations (e.g. time step width, calculation time, range of output, and selection of spatial interpolation). Right-end list gives numerical analysis results after the "Execute" command is selected from the menu. Clicking a XML file name on a list activates the editor or spreadsheet software for editing.

Numerical analysis results were first converted into a data format for input to post processing. Converted files were then selected from the post processing screen and the Draw Graph button was pressed to create a graph of time-series variation with pressure head at all incidental facilities in the hydraulic unit. The Output File button was pressed to convert the graphs into JPEG image files. Image files were created semi-automatically this way. Consequently, we can see the entire pipeline on a bird's eye view of Google Earth [8] and determine the maximum water hammer pressure. Clicking a place mark for incidental facilities displays a graph of time-series variation with pressure head at the position. Therefore, the transient hydraulic phenomenon at respective incidental facilities can be verified in detail.

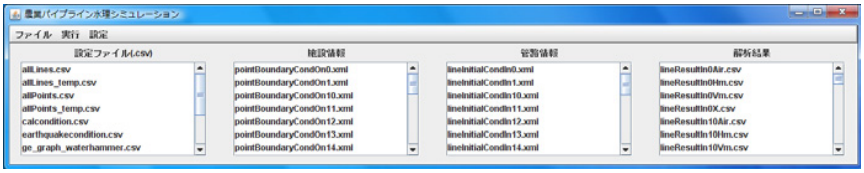

Fig.6. Windows screen of numerical analysis program

4. Conclusions

We developed design support system to create files based on ontology (ISML) and KML for numerical hydraulic analysis of irrigation pipelines. ISML is the standard for our proposed data management technique and KML is the standard for Google Earth. Consequently, the following results were obtained:

(1) Data creation work for numerical hydraulic analysis on pipelines can be reduced by using the Google Earth map display and input functions.

(2) A series of tasks, such as data input, numerical analysis and result visualization can be executed smoothly for a wide range of pipeline scenarios with large information volumes.

(3) Data input work can be simplified even for complicated pipeline systems that are difficult to enter with text file descriptions.

References

1. R. Mizoguchi, Ontology engineering, ohmusha, 7-14, (in Japanese), (2005)
2. S. Minegishi, M. Ishibashi, N. Fukuta, T Iijima and T. Yamaguchi, Supporting Software Engineering Processes with Ontologies, The 19th Annual Conference of the Japanese society for artificial intelligence, 1-4,
3. Ministry of Agriculture, Forestry and Fisheries, Rural Development Bureau, Rural Infrastructure Department, Design Division, Planning and design standards and operation of land improvement projects, commentary, and design, "Pipelines", 322-378 (2009)
4. Y. Tanaka, Proposal of development and maintenance management method based on object orientation of water hammer analysis program, 1-11, 284, IDRE journal
5. T.Asami, Web development using java/ XML, peason education, 3-108, (in Japanese), (2001)
6. M. Murata, XML［I］-XML Schema and RELAX-, The journal of institute of electronics information and communication engineers, 84,12,890-894 (2001)
7. Wiggert D.C. and Sundquist M.J. , Fixed-Grid Characteristics for Pipeline Transients, Journal of Hydraulics Division, ASCE, Vol.103, No. HY12, 1403-1416, (1977)
8. R. Sgrillo, http://www.sgrillo.net/googleearth/ index .htm